4

Langkah 1: Menemukan & Menganalisis Fakta

4.1 PENDAHULUAN

Bab ini menjelaskan langkah pertama dari perancangan basis data. Langkah ini meliputi pengumpulan dan penganalisisan kebutuhan pemakai yang dilakukan dengan tahapan sebagai berikut.

1. Mengidentifikasi bagian organisasi yang akan didukung oleh aplikasi basis data.
2. Mengidentifikasi pemakai utama basis data.
3. Melakukan pencarian informasi atau penemuan fakta.
4. Merangkum hasil penemuan fakta.

Output langkah ini adalah sebuah dokumen yang berisikan penjelasan rinci tentang kebutuhan data, kebutuhan transaksi, dan kebutuhan sistem dari pemakai basis data. Dokumen ini disebut dengan **Dokumen Spesifikasi Kebutuhan Pemakai** (Users' Requirement Specification Document).

4.2 IDENTIFIKASI BAGIAN ORGANISASI

Tujuan langkah ini adalah mengidentifikasi bagian organisasi yang akan didukung oleh aplikasi basis data. Misalnya, jika basis data yang dibuat akan menyimpan data dokter, pasien, obat-obatan, dan tindakan medis, maka bagian rumah sakit yang akan didukung adalah Bidang Medik, Bidang Keperawatan, Bidang Rekam Medik, dan Unit Farmasi.

Pengidentifikasian dapat dilakukan dengan melihat struktur organisasi, dokumentasi organisasi, dan/atau menanyakan langsung kepada pemakai.

4.3 IDENTIFIKASI PEMAKAI UTAMA

Tujuan langkah ini untuk menentukan pemakai pada bagian organisasi yang telah diidentifikasi pada langkah sebelumnya. Untuk setiap bagian organisasi, tentukan pemakai atau personil yang memiliki kepentingan terhadap aplikasi basis data yang akan dibuat. Personil tersebut dapat berupa orang-orang yang memiliki jabatan tertentu atau para staf.

Contoh pemakai pada basis data universitas adalah dosen, mahasiswa, ketua jurusan/program studi, dan staf bagian administrasi akademik & kemahasiswaan (BAAK), sedangkan contoh pemakai pada basis data rumah sakit adalah dokter, perawat, apoteker, dan kasir.

4.4 PENEMUAN FAKTA

Tujuan langkah ini untuk melakukan penemuan fakta di lapangan terkait dengan kebutuhan pemakai terhadap data, transaksi, dan sistem pada saat pemakai menggunakan aplikasi basis data.

Metode penemuan fakta yang dapat digunakan adalah sebagai berikut:

- Pengajian dokumen terkait. Metode ini akan membantu dalam menemukan alasan kenapa basis data perlu dibangun, memahami bagian organisasi terkait dengan pembangunan basis data,

dan memahami keadaan sistem yang sedang berjalan saat ini. Dokumen yang dikaji dapat berupa rencana strategis organisasi, bagan organisasi, dan dokumentasi aplikasi yang digunakan.

- Interview. Metode ini meliputi teknik penemuan fakta yang sering digunakan dalam pengumpulan informasi dengan cara bertanya langsung kepada pemakai. Teknik ini digunakan untuk membuktikan dan mengklarifikasi fakta, mengidentifikasi kebutuhan pemakai terhadap aplikasi yang akan dibuat, dan mengumpulkan ide dari pemakai.
- Observasi kegiatan di lapangan. Teknik ini dilakukan untuk memahami prosedur atau pekerjaan pemakai yang akan didukung oleh aplikasi basis data yang akan dibuat.
- Penyebaran kuesioner. Teknik ini dilakukan untuk mengumpulkan informasi dari para pemakai yang berjumlah besar.

Dengan menggunakan satu atau lebih teknik penemuan fakta di atas, dan menjadikan para (calon) pemakai dan sistem yang sudah ada sebagai sumber informasi, galilah informasi berikut:

1. Sasaran dari misi pembangunan basis data (mission objective).
2. Kebutuhan data yang akan disimpan pada basis data (data requirements).
3. Kebutuhan transaksi terhadap data yang akan disimpan pada basis data (transaction requirements).
4. Kebutuhan sistem basis data yang akan dibuat (system requirements).

Perlu diperhatikan bahwa pada langkah ini tercakup pengidentifikasian view pemakai. **View pemakai** mendefinisikan hal-hal yang dibutuhkan oleh sistem basis data dari sudut pandang jabatan/tugas pemakai tertentu

(misalnya dokter, apoteker, dan pasien) atau dari sudut pandang pemakaian aplikasi (misalnya aplikasi riwayat penyakit pasien dan aplikasi pengelolaan obat-obatan). Kebutuhan pemakai ini dapat berbeda satu sama lain atau dapat juga terjadi overlap. Karena bisa terdapat lebih dari satu jenis jabatan dan jenis pemakaian aplikasi, maka sistem basis data dapat memiliki lebih dari satu view pemakai.

4.4.1 Penggalian Sasaran Misi

Tanyakanlah misi kepada pemilik proyek pembangunan basis data dan galilah sasaran dari misi tersebut. Untuk penggalian sasaran misi, lakukan interview kepada setiap pemakai terpilih menggunakan beberapa contoh pertanyaan terbuka berikut:

- Deskripsi pekerjaan Anda?
- Pekerjaan apa saja yang Anda lakukan pada setiap hari?
- Data apa saja yang dilibatkan pada pekerjaan Anda tersebut?
- Jenis laporan apa saja yang Anda gunakan?
- Apa saja yang selalu ingin Anda monitor?
- Pelayanan apa saja yang diberikan oleh perusahaan atau organisasi Anda kepada pelanggan/tamu?

Contoh 4.1: Wawancara dan Penggalian Sasaran Misi dengan Seorang Dokter di Rumah Sakit

- Deskripsi pekerjaan Anda?

  "Pekerjaan saya adalah memeriksa dan mendiagnosa penyakit pasien, baik pasien rawat jalan maupun pasien rawat inap. Pemeriksaan harus dilakukan secara cepat dan tepat."

- Pekerjaan apa saja yang Anda lakukan pada setiap hari?

"Dari jam 06.00 s.d. 09.00 saya memeriksa pasien yang sedang dirawat. Dari hasil pemeriksaan, saya akan memutuskan apakah pasien tersebut tidak perlu dirawat lagi, perlu pengurangan/ penambahan obat, dan/atau perlu dilakukan tindakan medis lainnya. Dari jam 09.00 s.d. jam 17.00 saya melayani pasien rawat jalan. Saya memberikan resep obat, melakukan rekam medis, dan/atau memberikan nasihat kepada pasien untuk tindakan perawatan dan pencegahan."

- Data apa saja yang dilibatkan pada pekerjaan Anda tersebut?

  "Data riwayat penyakit pasien, data tindakan medis terhadap pasien, data obat yang telah diberikan kepada pasien, dan data jenis obat yang tersedia di rumah sakit."

- Jenis laporan apa saja yang Anda gunakan?

  "Jumlah pasien yang telah dirawat beserta tindakan medis yang telah dilakukan."

- Apa saja yang selalu ingin Anda monitor?

  "Tindakan medis pasien yang sedang dirawat di rumah sakit yang dalam penanganan saya."

- Pelayanan apa saja yang diberikan oleh perusahan/organisasi Anda kepada pelanggan/tamu?

  "Melaksanakan pelayanan medis (khusus, penunjang, dan tambahan), melaksanakan pelayanan rujukan kesehatan, dan melaksanakan pelayanan rawat inap dan rawat jalan."

Dari potongan hasil wawancara di atas, dapat disimpulkan bahwa *sasaran misi* pembangunan basis data pada rumah sakit tersebut, di antaranya sebagai berikut:

- Mengelola data pasien

- Mengelola data obat
- Mengelola data tindakan medis
- Melakukan pencarian data pasien
- Melakukan pencarian data obat
- Melakukan pencarian data tindakan medis
- Memonitor tindakan medis yang telah dilakukan kepada pasien
- Melaporkan jumlah pasien yang telah ditangani
- Melaporkan tindakan medis yang telah diberikan

Perhatikan bahwa pengelolaan data meliputi aktivitas rekam, ubah, dan hapus data. Contoh di atas adalah sasaran misi yang didapat dari hasil wawancara pada satu orang dokter. Idealnya penggalian sasaran misi sebaiknya dilakukan berdasarkan hasil wawancara dari berbagai jenis kelompok pemakai, dan menyimpulkan hasil wawancara tersebut menjadi sasaran misi dalam format seperti contoh di atas. ■

4.4.2 Penggalian Kebutuhan Data

Temukan data yang menjadi perhatian utama pemakai dari sasaran misi pembangunan basis data yang telah dibuat pada langkah sebelumnya. Untuk setiap data (*d*), ajukan beberapa pertanyaan terbuka berikut untuk meng-gali lebih jauh atribut data yang akan disimpan pada basis data.

- Jenis data/informasi apa saja yang Anda ingin ketahui tentang *d*?
- Apa saja yang Anda lakukan terhadap data/informasi pada *d*?

Contoh 4.2: Wawancara dan Penggalian Kebutuhan Data

Dari sasaran misi pada contoh di atas, maka data yang menjadi fokus utama adalah pasien, obat, dan tindakan medis. Untuk masing-masing

data, tanyakan lebih lanjut kepada pemakai (pada contoh ini dokter) tentang jenis informasi dan perlakuan terhadap data. Di bawah ini contoh hasil wawancara untuk data pasien:

- Jenis data/informasi apa saja yang Anda ingin ketahui tentang pasien?

  "Nama, umur, berat badan, tinggi badan, tekanan darah, denyut nadi, pekerjaan, dan riwayat penyakit pasien."

- Apa saja yang Anda lakukan terhadap data/informasi pada pasien?

  "Saya perlu untuk merekam data umur, berat badan, tinggi badan, tekanan darah, denyut nadi, dan hasil diagnosa penyakit pasien. Saya juga perlu menghapus atau menandai pasien yang telah meninggal." ■

4.4.3 Penggalian Kebutuhan Transaksi

Kebutuhan transaksi didefinisikan berdasarkan hasil wawancara pada dua langkah terdahulu. Kebutuhan transaksi dikelompokkan atas tiga jenis, yaitu transaksi perekaman data, transaksi pengubahan/penghapusan data, dan transaksi pengambilan data.

▌Contoh 4.3: Penggalian Kebutuhan Transaksi

Berikut contoh kebutuhan transaksi yang diambil dari contoh sebelumnya.

Transaksi Perekaman Data:

- Rekam/entri data detail pasien
- Rekam/entri data detail obat
- Rekam/entri data detail tindakan medis

Transaksi Pengubahan/Penghapusan Data:

- Ubah/hapus data detail pasien
- Ubah/hapus data detail obat
- Ubah/hapus data detail tindakan medis

Transaksi Pengambilan Data:

- Tampilkan data detail pasien tertentu
- Tampilkan data detail pasien yang dirawat oleh dokter tertentu
- Tampilkan data detail tindakan medis pasien tertentu yang dirawat oleh dokter tertentu
- Tampilkan data detail obat untuk penyakit tertentu
- Tampilkan jumlah pasien yang telah dirawat oleh dokter tertentu ■

4.4.4 Penggalian Kebutuhan Sistem

Penggalian kebutuhan sistem bertujuan untuk menggali informasi umum yang dibutuhkan oleh sistem serta spesifikasinya. Pertanyaan ini dapat mengacu pada kebutuhan transaksi yang telah didapat pada langkah sebelumnya.

Contoh dari beberapa pertanyaan yang dapat diajukan untuk menggali kebutuhan sistem adalah sebagai berikut:

- Transaksi apa yang sering dijalankan pada basis data?
- Transaksi apa yang kritis terhadap beroperasinya perusahaan/ organisasi?
- Kapan transaksi kritis tersebut dijalankan?

- Kapan waktu beban kerja rendah, sedang, dan tinggi dari transaksi kritis tersebut berlangsung?
- Jenis keamanan apa yang diinginkan pada sistem basis data?
- Apakah ada data penting yang hanya boleh diakses oleh orang-orang tertentu saja?
- Data histori apa saja yang ingin disimpan?

Penggalian kebutuhan sistem juga meliputi penggalian spesifikasi sistem, yang di antaranya meliputi: ukuran awal basis data, laju pertumbuhan ukuran basis data, jenis dan rata-rata jumlah pencarian tuple, dan performa sistem yang diharapkan.

Contoh 4.4: Hasil Penggalian Kebutuhan Sistem

Berikut contoh kebutuhan sistem terhadap transaksi yang diambil dari contoh sebelumnya:

- Transaksi apa yang sering dijalankan pada basis data?

  "Menampilkan info detail pasien tertentu dan info detail obat untuk penyakit tertentu."

- Transaksi apa yang kritis terhadap beroperasinya perusahaan/ organisasi?

  "Menampilkan info tindakan medis oleh dokter tertentu."

- Kapan transaksi kritis tersebut dijalankan?

  "Setiap hari."

- Kapan waktu beban kerja rendah, sedang, dan tinggi dari transaksi kritis tersebut berlangsung?

  "Waktu yang tersibuk adalah waktu pemeriksaan pasien rawat inap, yaitu dari jam 06.00 s.d. 08.00. Sedangkan untuk waktu

normal dan rendah, masing-masing adalah waktu pemeriksaan pasien rawat jalan (09.00 s.d. 17.00) dan waktu setelah praktik dokter (17.00 s.d. 08.00)."

Berikut contoh spesifikasi sistem yang dibutuhkan:

- Ukuran awal basis data
  - Ada sekitar 1000 orang pasien yang terdaftar.
  - Ada sekitar 500 jenis obat-obatan yang digunakan di rumah sakit.
  - Ada sekitar 3000 tindakan medis yang telah dilakukan.
- Laju pertumbuhan ukuran basis data
  - Rata-rata 5 pasien baru yang mendaftar setiap hari atau 150 pasien baru per bulan. Pasien yang meninggal akan dihapus dari basis data. Terdapat rata-rata 10 kali penghapusan pasien dalam sebulan.
  - Rata-rata 10 tindakan medis yang dilakukan pada setiap hari atau 300 tindakan medis per bulan.
  - Rata-rata 5 jenis obat baru yang didaftarkan pada setiap bulan. Obat yang tidak digunakan lagi akan dihapus dari basis data. Terdapat rata-rata 2 kali penghapusan obat lama pada setiap bulannya.
- Jenis dan rata-rata jumlah pencarian tuple
  - Pencarian data detail pasien tertentu: rata-rata 150 kali per hari.
  - Pencarian data detail pasien yang dirawat dokter tertentu: rata-rata 100 kali per hari.

- Pencarian data detail tindakan medis pasien tertentu yang dirawat oleh dokter tertentu: rata-rata 25 kali per hari.
    - Tampilkan data detail obat penyakit tertentu: rata-rata 200 kali per hari.
    - Tampilkan jumlah pasien yang telah dirawat oleh dokter tertentu: rata-rata 50 kali per bulan.
- Performa sistem yang diharapkan
    - Pada jam operasional rumah sakit (06.00 s.d. 17.00) yang bukan jam tersibuk:
        - Waktu respons sistem yang diharapkan pada saat pencarian satu tuple adalah di bawah 2 detik.
        - Waktu respons sistem yang diharapkan pada saat pencarian beberapa tuple adalah di bawah 5 detik.
        - Waktu respons sistem yang diharapkan pada setiap update atau penyimpanan data adalah di bawah 2 detik.
    - Pada jam operasional rumah sakit (06.00 s.d. 17.00) yang merupakan jam tersibuk:
        - Waktu respons sistem yang diharapkan pada saat pencarian satu tuple adalah di bawah 5 detik.
        - Waktu respons sistem yang diharapkan pada saat pencarian beberapa tuple adalah di bawah 10 detik.
        - Waktu respons sistem yang diharapkan pada setiap update atau penyimpanan data adalah di bawah 5 detik. ■

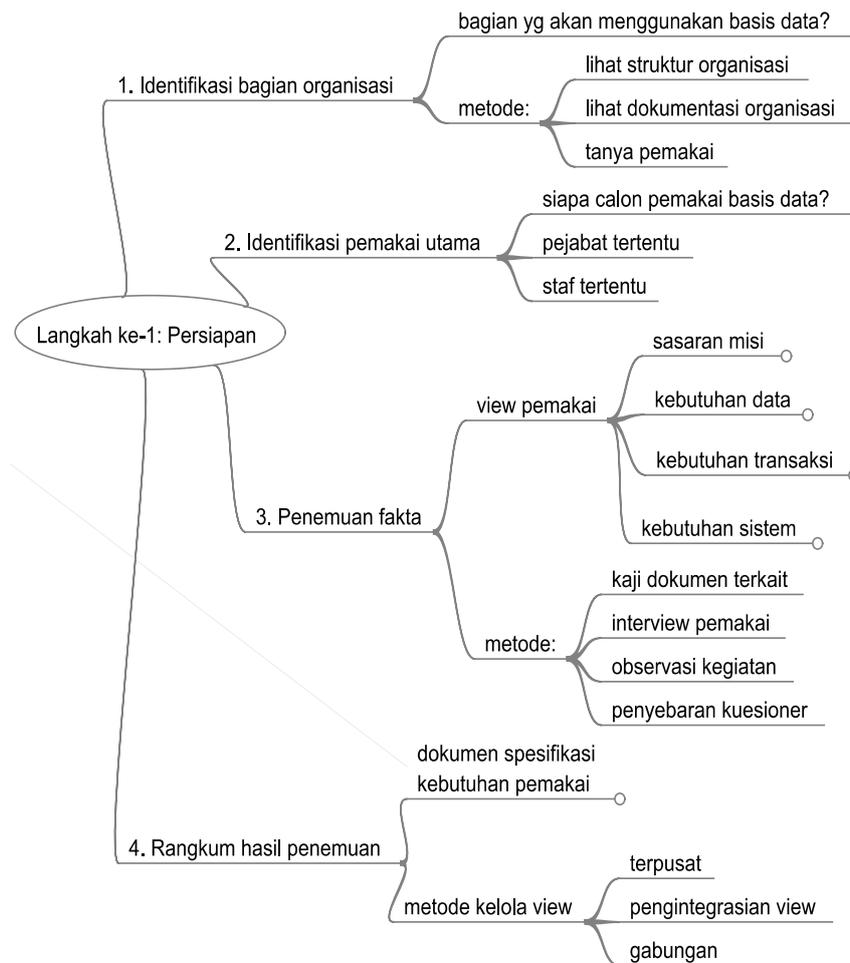
4.5 RANGKUM HASIL PENEMUAN FAKTA

Dari langkah sebelumnya, didapatkan satu atau lebih view pemakai yang menjelaskan tentang sasaran misi, kebutuhan data, kebutuhan transaksi, dan kebutuhan sistem. Terdapat tiga pendekatan untuk mengelola view pemakai yang jamak ini:

- **Pendekatan Terpusat** (Centralized Approach) yang menggabungkan semua view pemakai menjadi satu view pemakai besar, dan kemudian membangun data model (entity relationship diagram) terhadap view pemakai besar tersebut.
- **Pendekatan Pengintegrasian View** (View Integration Approach) yang membangun data model lokal untuk masing-masing view pemakai, dan kemudian menggabungkan data-data model lokal tersebut menjadi satu data model global setelah melakukan normalisasi tabel (yaitu setelah melakukan langkah ke-4).
- **Pendekatan Gabungan** (Hybrid Approach) dengan cara sebagai berikut:
    1. Mengelompokkan view-view pemakai atas beberapa kelompok,
    2. Untuk setiap kelompok, gabungkan semua view pemakai pada kelompok tersebut menjadi satu view pemakai kelompok,
    3. Membangun data model untuk masing-masing view pemakai kelompok, dan
    4. Menggabungkan data model view pemakai kelompok menjadi satu data model global setelah melakukan normalisasi tabel.

Pendekatan yang digunakan pada buku ini adalah pendekatan terpusat. Oleh karena itu, pada langkah ini rangkumlah hasil penemuan menjadi satu view pemakai global yang menghasilkan sebuah dokumen spesifikasi kebutuhan pemakai.

Selain menuliskan visi (tujuan utama) dan misi (kegiatan tertentu untuk mencapai visi) pembangunan basis data, rangkuman juga menuliskan tiga sudut pandang, yaitu: kebutuhan data, kebutuhan transaksi, dan kebutuhan sistem. Contoh dari dokumen spesifikasi kebutuhan pemakai dapat dilihat pada Bab 11 Studi Kasus.

4.6 RINGKASAN MINDMAP

5

Langkah 2: Membuat ERD

5.1 PENDAHULUAN

Bab ini menjelaskan pembuatan Diagram Hubungan Entitas atau yang dikenal dengan sebutan Entity Relationship Diagram (ERD). Pembuatan ERD dilakukan berdasarkan dokumen spesifikasi kebutuhan pemakai yang merupakan output dari langkah sebelumnya. ERD merupakan model atau abstraksi data yang merupakan fokus utama suatu organisasi. Pembuatan ERD dilakukan sebagai berikut.

1. Tentukan jenis entitas utama pada organisasi.
2. Tentukan jenis hubungan utama antar-jenis entitas.
3. Tentukan multiplicity jenis hubungan entitas.
4. Tentukan atribut atau properti dari jenis entitas dan jenis hubungan entitas.
5. Buat ERD dengan menggambarkan jenis entitas, hubungannya, serta atributnya dengan notasi yang dipilih.
6. Periksa dan sempurnakan struktur ERD.

Penulisan ERD dapat dilakukan menggunakan beberapa notasi di antaranya notasi UML, Crow's Foot, dan Chen. Buku ini menggunakan notasi UML karena notasi UML merupakan bahasa pemodelan umum dan standar untuk pengembangan perangkat lunak. Meskipun demikian, buku ini juga memberikan penjelasan notasi Crow's Foot dan Notasi Chen pada Lampiran 1.

Perhatikan bahwa karena perbedaan metode penemuan fakta yang digunakan dan subjektivitas yang tidak bisa dihindari dalam penentuan jenis entitas, jenis hubungan entitas, dan atribut, maka ERD yang dibuat dapat berbeda dari satu perancang basis data dengan perancang basis data yang lain. Yang penting pastikan bahwa ERD tersebut bebas dari permasalahan struktur (Subbab 5.7) dan dapat memenuhi semua kebutuhan pemakai (Subbab 7.4).

5.2 TENTUKAN JENIS ENTITAS

Jenis Entitas (Entity Type) adalah sekumpulan objek-objek entitas yang memiliki karekteristik yang sama dan memiliki keberadaan yang bebas.

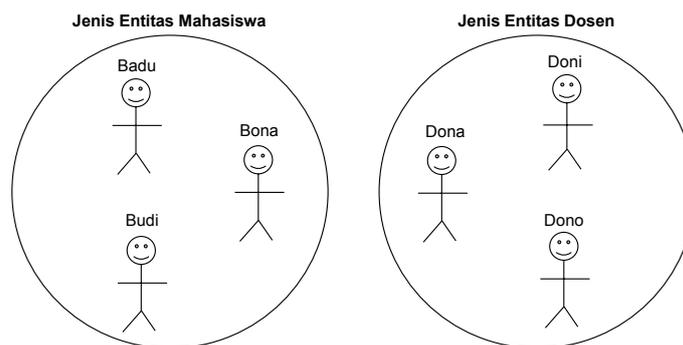
Objek Entitas (Entity Occurence) adalah sebuah objek dari suatu jenis entitas yang dapat diidentifikasi secara unik.

Jenis entitas merepresentasikan "sekumpulan objek" pada dunia nyata yang memiliki properti atau karekteristik yang sama. Jenis entitas dapat terdiri atas objek-objek entitas yang memiliki keberadaan secara fisik maupun abstrak. Sebagai contoh, pada (dunia nyata) rumah sakit dapat ditetapkan jenis entitas fisik, yaitu dokter, perawat, dan pasien, dan jenis entitas abstrak, yaitu penyakit dan pelayanan.

Contoh 5.1: Jenis Entitas dan Objek Entitas

Gambar 5.1 memperlihatkan dua jenis entitas, yaitu jenis entitas Mahasiswa dan Dosen. Objek-objek entitas Badu, Bona, dan Budi pada

jenis entitas Mahasiswa memiliki properti yang sama, yaitu atribut nama, nim, dan jenis kelamin, sedangkan objek-objek entitas Doni, Dona, dan Dono pada jenis entitas Dosen juga memiliki atribut yang sama, yaitu atribut nama dan nidn. Setiap objek entitas mahasiswa dan dosen masing-masing dapat diidentifikasi secara unik dengan melihat nilai atribut nim dan nidn. ■

Lebih jauh, jenis entitas dapat diklasifikasikan sebagai **Jenis Entitas Kuat** (Strong Entity Type) dan **Jenis Entitas Lemah** (Weak Entity Type). Jenis entitas kuat memiliki keberadaan yang tidak bergantung pada keberadaan jenis entitas lain, sedangkan jenis entitas lemah memiliki keberadaan yang bergantung pada keberadaan jenis entitas lain.

Gambar 5.1 Contoh jenis entitas dan objek entitas

Contoh 5.2: Jenis Entitas Kuat dan Lemah

Berikut contoh jenis entitas kuat dan entitas lemah yang saling berhubungan:

- Keberadaan jenis entitas kuat BukuTeks tidak bergantung pada jenis entitas lain, termasuk pada jenis entitas EdisiBuku. Sebaliknya, keberadaan jenis entitas lemah EdisiBuku sangat bergantung pada keberadaan jenis entitas BukuTeks; Jika tidak ada buku teks maka edisi buku tersebut juga tidak akan ada.

- Keberadaan jenis entitas kuat Pelanggan tidak bergantung pada jenis entitas lain, termasuk pada jenis entitas PembayaranHutang. Sebaliknya, keberadaan jenis entitas lemah PembayaranHutang sangat bergantung pada keberadaan jenis entitas Pelanggan; Pembayaran hutang tidak akan pernah ada jika tidak ada pelanggan yang berhutang. ■

5.2.1 Langkah Pengidentifikasian

Jenis entitas diidentifikasi dengan membaca dan memahami dokumen spesifikasi kebutuhan pemakai, sebagai berikut:

1. Temukan <u>kata benda</u> dan <u>frase kata benda</u> yang menjadi perhatian utama pemakai. Kata benda dan frase kata benda yang ditemukan berpotensi menjadi jenis entitas utama.
2. Tuliskan jenis entitas pada kamus data dengan menyertakan informasi berupa nama, deskripsi, nama alias (jika ada), dan jumlah objek entitas dari masing-masing jenis entitas tersebut. Agar mudah dipahami, tuliskan kamus data dalam format tabel seperti di bawah ini (contoh dapat dilihat pada Subbab 11.3).

Jenis Entitas	Deskripsi	Nama Alias	Jumlah Objek Entitas

5.2.2 Notasi UML

Notasi UML untuk jenis entitas adalah empat persegi panjang yang diberi label dengan nama jenis entitas pada notasi tersebut. Jika nama tersebut terdiri atas dua kata atau lebih, penulisannya tidak menggunakan spasi di antara kata, dan setiap huruf awal dimulai dengan huruf besar. Gambar 5.2 memperlihatkan contoh penulisan notasi jenis entitas.

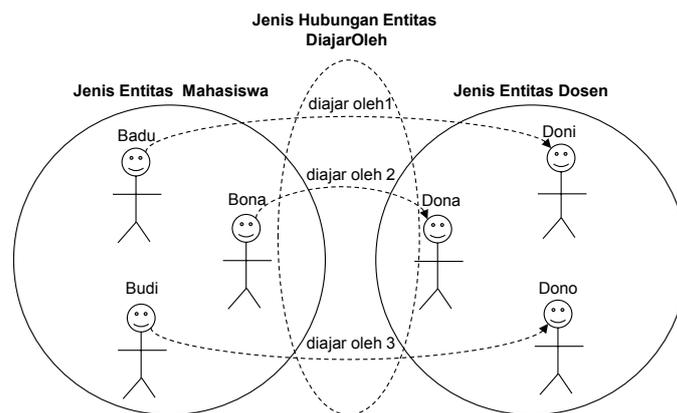
Gambar 5.2 Jenis entitas Mahasiswa dan MataKuliah

5.3 TENTUKAN JENIS HUBUNGAN ENTITAS

Jenis Hubungan Entitas (Relationship Type) adalah sekumpulan asosiasi atau hubungan di antara objek entitas yang menjadi fokus utama pada perancangan basis data.

Objek Hubungan Entitas (Relationship Type Occurrence) adalah objek dari asosiasi atau jenis hubungan entitas yang dapat diidentifikasi secara unik.

Jenis hubungan entitas merepresentasikan "sekumpulan hubungan di antara objek-objek entitas" (pada dunia nyata), di mana hubungan tersebut memiliki properti atau karekteristik yang sama. Setiap jenis hubungan entitas memiliki nama yang mendeskripsikan fungsi dari jenis hubungan entitas tersebut. Setiap jenis hubungan entitas juga memiliki arah dan setiap hubungan dapat dilihat (dibaca) dari dua arah yang berlawanan.

Contoh 5.3: Jenis dan Objek Hubungan Entitas

Gambar 5.3 Contoh jenis dan objek hubungan entitas

Gambar 5.3 memperlihatkan contoh jenis hubungan entitas DiajarOleh yang mengasosiasikan jenis entitas Mahasiswa dan Dosen. Ada tiga objek hubungan entitas, yaitu "diajar oleh 1" yang menghubungkan Badu-Doni, "diajar oleh 2" yang menghubungkan Bona-Dona, dan "diajar oleh 3" yang menghubungkan Budi-Dono.

Jenis hubungan entitas DiajarOleh dapat dilihat dari arah berlawanan, yaitu dari jenis entitas Dosen ke Mahasiswa dengan nama jenis hubungan entitas Mengajar (Dosen Mengajar Mahasiswa). ■

5.3.1 Langkah Pengidentifikasian

Sama seperti jenis entitas, jenis hubungan entitas diidentifikasi dengan membaca dan memahami dokumen spesifikasi kebutuhan pemakai, sebagai berikut.

1. Temukan kata kerja atau frase kata kerja yang menjadi perhatian utama pemakai.
2. Untuk setiap jenis entitas yang telah diidentifikasi pada langkah sebelumnya, periksa apakah terdapat hubungan yang menjadi fokus utama pemakai, baik secara eksplisit maupun secara implisit.
3. Tuliskan jenis hubungan entitas pada kamus data dengan menyertakan informasi berupa nama-nama jenis entitas beserta nama jenis hubungan entitas yang mengasosiasikan jenis entitas tersebut. Contoh penulisan ini dapat dilihat pada Subbab 11.3.

Jenis Entitas	Jenis Hubungan	Jenis Entitas

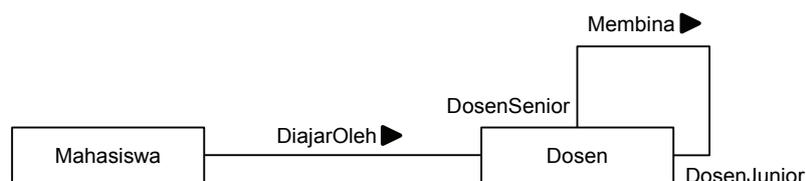
Umumnya, jenis hubungan entitas menghubungkan dua jenis entitas. Jenis hubungan entitas ini disebut dengan **Jenis Hubungan Entitas Biner**.

Misalnya, jenis hubungan entitas DiajarOleh pada Gambar 5.3 merupakan jenis hubungan entitas biner karena jenis hubungan entitas ini mengasosiasikan dua jenis entitas, yaitu Dosen dan Mahasiswa.

Bentuk jenis hubungan entitas lain adalah jenis hubungan entitas tunggal (recursive/unary relationship type) dan jenis hubungan entitas kompleks (complex relationship type). **Jenis Hubungan Entitas Tunggal** menghubungkan satu jenis entitas yang sama, sedangkan **Jenis Hubungan Entitas Kompleks** menghubungkan lebih dari dua jenis entitas. Misalnya, jenis hubungan entitas Pembinaan di antara sesama dosen pada Gambar 5.4 merupakan contoh jenis hubungan entitas tunggal, di mana seorang dosen (senior) dapat membina satu atau beberapa dosen (junior).

5.3.2 Notasi UML

Notasi UML untuk jenis hubungan entitas adalah garis yang menghubungkan jenis entitas, dan garis tersebut diberi label kata kerja (transitif atau intransitif) dengan disertai mata panah sebagai penunjuk arah hubungan. Jika nama jenis hubungan entitas terdiri atas dua kata atau lebih, maka nama tersebut dituliskan tanpa menggunakan spasi pemisah di antara kata, dan huruf awal setiap kata dituliskan dengan huruf besar.

Contoh 5.4: Notasi Jenis Hubungan Entitas

Gambar 5.4 Jenis hubungan entitas DiajarOleh dan Membina

Gambar 5.4 memperlihatkan contoh jenis hubungan entitas Biner DiajarOleh yang mengasosiasikan jenis entitas Mahasiswa dan Dosen,

serta jenis hubungan entitas tunggal Membina yang secara berulang menghubungkan satu jenis entitas Dosen.

Jenis entitas Dosen memiliki **Nama Peran** (role name) DosenSenior dan DosenJunior pada kedua ujung garis. Fungsi nama peran adalah memberikan penjelasan lebih lanjut tentang jenis hubungan entitas. ERD ini dapat dibaca sesuai dengan arah hubungan sebagai berikut: "Mahasiswa diajar oleh dosen dan dosen senior dapat membina dosen junior". ■

5.4 TENTUKAN MULTIPLICITY

Multiplicity adalah deskripsi tentang jumlah suatu objek entitas yang mungkin dapat diasosiasikan dengan satu objek entitas lain.

Pada contoh jenis hubungan entitas DiajarOleh pada Gambar 5.4, satu objek entitas Mahasiswa dapat diajar oleh satu atau beberapa objek entitas Dosen, dan satu objek entitas dosen dapat mengajar satu atau lebih objek entitas mahasiswa.

Multiplicity merupakan suatu ketentuan (constraint) yang mengatur hubungan di antara objek-objek entitas. Oleh karena itu, multiplicity merupakan kebijakan atau aturan bisnis (business rule) yang ditetapkan oleh pemakai atau organisasi, yang dapat berbeda dari satu organisasi dengan organisasi lain.

Contoh 5.5: Multiplicity Jenis Hubungan Entitas Merawat

Misalnya, suatu rumah sakit dapat memiliki kebijakan bahwa seorang dokter hanya dapat merawat maksimum 10 orang pasien per hari, sedangkan kebijakan pada rumah sakit lain mungkin berbeda, yaitu maksimum 15 orang pasien per hari. ■

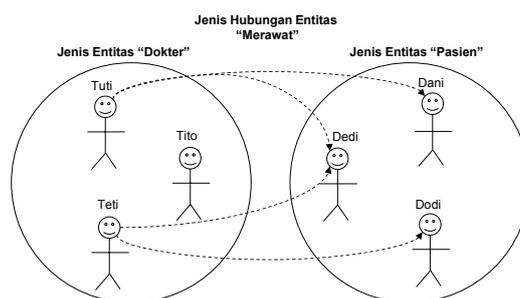
Lebih lanjut, multiplicity dapat dibagi menjadi dua batasan atau aturan, yaitu:

1. **Batasan Kardinalitas** (Cardinality Constraint) yang menentukan jumlah maksimum objek entitas lain yang dapat berasosiasi dengan suatu objek entitas tertentu.
2. **Batasan Partisipasi** (Participation Constraint) yang menentukan apakah semua atau sebagian objek entitas berpartisipasi pada suatu jenis hubungan entitas.

Partisipasi dikatakan **Wajib** (mandatory atau full) jika semua objek entitas berpartisipasi pada jenis hubungan entitas. Dan partisipasi dikatakan **Opsional** atau **Partial**, jika tidak semua objek entitas berpatisipasi pada jenis hubungan entitas.

Contoh 5.6: Batasan Kardinalitas dan Partisipasi

Misalnya, pada jenis hubungan entitas Merawat, seorang dokter dapat merawat 0 pasien (atau tidak merawat satu orang pasien pun) dan maksimum lebih dari 1 pasien (banyak). Pada contoh ini, batasan kardinalitas jenis hubungan entitas Merawat terhadap jenis entitas dokter adalah banyak, dan batasan partisipasinya bersifat opsional (Gambar 5.5).

Dokter Tuti dan Teti masing-masing merawat dua orang pasien (banyak), sedangkan Dokter Tito tidak merawat satu orang pasien pun. Pada contoh ini, Dokter Tito tidak berpartisipasi dalam jenis hubungan entitas Merawat karena dia tidak terasosiasi pada seorang pasien pun. ■

Gambar 5.5 Contoh batasan kardinalitas dan partisipasi

Karena jenis hubungan entitas dapat dilihat dari dua arah berlawanan, maka batasan kardinalitas dan partisipasi dapat juga diterapkan pada arah berlawanan. Misalnya pada contoh di atas, jika dilihat dari arah berlawanan, terdapat batasan bahwa seorang pasien wajib dirawat minimum oleh seorang dokter karena semua pasien mempunyai dokter yang merawatnya (batasan partisipasi bersifat wajib), dan maksimum dirawat oleh lebih dari satu orang dokter, yaitu pada kasus pasien Dedi (batasan kardinalitas banyak).

Perbandingan kardinalitas yang dilihat dari dua arah hubungan disebut dengan **Rasio Kardinalitas** (Cardinality Ratio) atau **Derajat Hubungan** (Relationship Degree). Pada contoh di atas, karena seorang dokter dapat merawat maksimum banyak pasien dan seorang pasien dapat dirawat maksimum oleh banyak dokter, maka rasio kardinalitasnya adalah many-to-many (N:N). Perhatikan bahwa nilai kardinalitas yang lebih besar dari satu dikatakan banyak (many) dengan simbol N[1].

Terdapat tiga jenis rasio kardinalitas, yaitu:

1. **One-to-One** (1:1), jika nilai kardinalitas pada kedua arah 1.
2. **One-to-Many** (1:N) atau **Many-to-One** (N:1), jika nilai kardinalitas pada salah satu arah 1 dan arah lainnya banyak (N).
3. **Many-to-Many** (N:N), jika nilai kardinalitas pada kedua arah banyak (N).

5.4.1 Langkah Pengidentifikasian

Pengidentifikasian multiplicity dapat dilakukan sebagai berikut:

1. Perhatikan penjelasan rinci setiap jenis hubungan entitas pada kamus data dan temukan nilai kardinalitas dan partisipasinya. Jika nilai tersebut tidak dapat ditemukan atau tidak lengkap,

[1] Simbol lain yang umumnya digunakan untuk many adalah '*' atau 'M'.

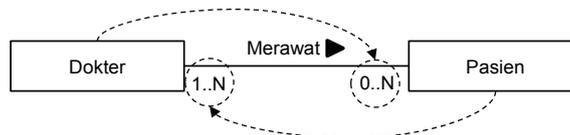
mintalah penjelasan lebih lanjut kepada pemakai, dan tambahkan informasi tersebut pada dokumen spesifikasi.

2. Dokumentasikan semua jenis hubungan entitas lengkap dengan multiplicity-nya dengan mengisi tabel berikut. Ini melengkapi kamus data dengan penambahan rasio kardinalitas pada tabel jenis hubungan entitas. Contoh penulisan ini dapat dilihat pada Subbab 11.3.

Jenis Entitas	Rasio Kardinalitas	Jenis Hubungan Entitas	Rasio Kardinalitas	Jenis Entitas

5.4.2 Notasi UML

Nilai multiplicity dituliskan di kedua ujung garis jenis hubungan entitas dalam bentuk "nilai_partisipasi..nilai_kardinalitas". Nilai partisipasi ditulis 0 untuk jenis hubungan opsional. Jenis hubungan entitas dapat dibaca <u>dalam dua arah</u> berdasarkan multiplicity tersebut.

Contoh 5.7: Multiplicity dan Rasio Kardinalitas

Gambar 5.6 Multiplicity jenis hubungan entitas Merawat

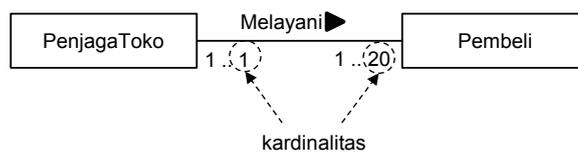
Gambar 5.6 memperlihatkan contoh jenis hubungan entitas Merawat yang mengasosiasikan jenis entitas Dokter dan Pasien, sebagai berikut:

1. Multiplicity dengan arah jenis entitas Dokter ke Pasien diletakkan pada ujung garis yang berseberangan dengan jenis entitas Dokter, dan dibaca "Seorang Dokter dapat merawat minimum 0 orang pasien dan maksimum banyak (N) pasien".

2. Multiplicity dengan arah jenis entitas Pasien ke Dokter diletakkan pada ujung garis yang berseberangan dengan jenis entitas Pasien dan dibaca: "Seorang Pasien dapat dirawat oleh minimum 1 orang dokter dan maksimum banyak (N) dokter".
3. Rasio kardinalitas jenis hubungan tersebut adalah many-to-many (N:N) yang didapat dari nilai kardinalitas di kedua ujung garis. ■

Nilai partisipasi dan kardinalitas dapat dituliskan dalam angka tertentu jika nilai minimum dan maksimum diketahui.

Contoh 5.8: Kardinalitas dalam Jumlah Tertentu

Perhatikan Gambar 5.7. Pada arah sesuai dengan mata panah dibaca "Seorang penjaga toko dapat melayani minimal 1 pembeli dan maksimal 20 pembeli", sedangkan dalam arah yang berlawanan dibaca "Seorang pembeli hanya dapat dilayani oleh 1 penjaga toko saja". ■

Gambar 5.7 Multiplicity jenis hubungan entitas Melayani

5.5 TENTUKAN ATRIBUT

Atribut (Attribute) adalah properti atau karekteristik dari jenis entitas dan jenis hubungan entitas.

Atribut menyimpan nilai-nilai yang mendeskripsikan objek entitas dan objek hubungan entitas. Nilai-nilai tersebut merupakan bagian utama data yang akan disimpan dan dikelola pada basis data[2].

[2] Secara umum, atribut pada jenis entitas adalah sama dengan kolom pada tabel (lihat *Bab 6 Pemetaan ERD*).

Pada jenis entitas dan hubungan entitas yang diperlihatkan pada Gambar 1.1 dapat diberikan atribut-atribut, sebagai berikut:

- Jenis entitas Mahasiswa dideskripsikan oleh atribut nama, nim, jenis kelamin, dan kode mata kuliah.
- Jenis entitas Dosen dideskripsikan oleh atribut nama, nidn, dan kode mata kuliah.
- Jenis entitas Mata Kuliah dideskripsikan oleh atribut nama, kode, dan sks.
- Jenis hubungan entitas Mengajar dapat diberikan atribut tahun dan semester yang mendeskripsikan kapan seorang dosen mengajar suatu mata kuliah.

Data yang merupakan nilai-nilai atribut dari tiga orang mahasiswa, tiga orang dosen, serta tiga mata kuliah inilah yang merupakan data utama yang akan disimpan pada basis data seperti pada tabel-tabel di Gambar 1.1.

Lebih lanjut, atribut dapat dibagi atas beberapa jenis:

1. **Atribut Sederhana** (Simple Attribute), yaitu atribut yang memiliki komponen tunggal, seperti atribut umur dan jenis kelamin.
2. **Atribut Campuran** (Composite Attribute), yaitu atribut yang terdiri atas beberapa atribut sederhana sebagai komponen penyusunnya, seperti atribut alamat yang dapat terdiri atas sub-atribut jalan, nama kota, dan kode pos.
3. **Atribut Bernilai Tunggal** (Single-Valued Attribute), yaitu atribut yang hanya dapat menyimpan satu nilai pada setiap objek entitas dan objek hubungan entitas, misalnya atribut umur, nim, dan nidn.

4. **Atribut Bernilai Jamak** (Multi-Valued Attribute), yaitu atribut yang memiliki nilai lebih dari satu. Misalnya, jika seorang staf dapat memiliki dua nomor handphone, maka atribut tersebut pada jenis entitas staf merupakan atribut bernilai jamak.

5. **Atribut Turunan** (Derived Attribute), yaitu atribut yang nilainya dihitung berdasarkan nilai atribut atau komponen lain yang berhubungan dengan atribut tersebut. Misalnya, atribut umur pada jenis entitas Mahasiswa yang dihitung berdasarkan tanggal lahir dan tanggal saat ini.

6. **Atribut Kunci** (Key Attribute), yaitu atribut yang nilainya dapat membedakan secara unik objek entitas. Atribut kunci berhubungan dengan candidate key yang telah dijelaskan pada Bab 1. Misalnya, atribut nim dan nidn yang masing-masing terdapat pada jenis entitas Mahasiswa dan Dosen merupakan atribut kunci.

Atribut kunci dapat terdiri atas gabungan dua atribut atau lebih, di mana nilai atribut-atribut tersebut secara bersama-sama membedakan secara unik objek entitas. Gabungan atribut ini disebut dengan **Kunci Komposit** (composite key).

Contoh 5.9: Kunci Komposit

Grade

gradeId	*kodeMakul*	*nim*	*grade*	*idDosen*
gr001	mk001	mhs001	A	dsn02
gr002	mk001	mhs002	B	dsn02
gr003	mk002	mhs001	B	dsn01

Gambar 5.8 Kunci komposit pada objek entitas Grade

Gambar 5.8 memperlihatkan objek entitas Grade yang disimpan pada tabel. Setiap tuple mewakili satu objek entitas. Jenis entitas Grade memiliki lima atribut, yaitu gradeId, kodeMakul, nim, grade, dan idDosen pengajar. Salah satu atribut kunci pada jenis entitas Grade adalah atribut kodeMakul dan nim yang merupakan composite key. Nilai composite key ini secara bersama-sama dapat membedakan masing-masing objek entitas karena setiap nim pasti hanya memiliki satu grade untuk setiap mata kuliah yang telah diambil. Perhatikan bahwa atribut kodeMakul saja atau nim saja tidak dapat dijadikan atribut kunci karena masing-masing atribut memiliki nilai duplikat (atau tidak unik).

Suatu jenis entitas dapat memiliki lebih dari satu atribut kunci. Sekumpulan atribut kunci pada suatu jenis entitas disebut dengan kunci kandidat (candidate key). Pada jenis entitas Grade terdapat dua candidate key, yaitu (1) atribut gradeId, dan (2) atribut kodeMakul dan nim. ■

Jika menemukan lebih dari satu candidate key, pilihlah salah satu atribut kunci untuk dijadikan primary key, dan selebihnya sebagai kunci alternatif (alternate key). Jika hanya ada satu atribut kunci, maka atribut kunci tersebut satu-satunya pilihan untuk dijadikan primary key.

Berikut beberapa kriteria pemilihan atribut primary key dari candidate key yang ada.

- Pilihlah atribut kunci yang nilainya jarang berubah. Misalnya, atribut nip dan nim.
- Pilihlah atribut kunci yang selalu memiliki nilai yang valid (atau tidak akan pernah memiliki null). Misalnya, atribut nip dan nim yang selalu memiliki nilai atau tidak pernah kosong.
- Pilihlah atribut kunci yang memiliki jumlah atribut penyusun minimal. Misalnya, pada jenis entitas Grade di atas pilihlah

atribut gradeId daripada kodeMakul, dan nim sebagai atribut primary key.

- Pilihlah atribut kunci yang memiliki jumlah karakter yang minimum. Misalnya, pilihlah nim yang memiliki 10 karakter dibandingkan alamat yang memiliki 50 karakter (jika alamat unik).

Perhatikan bahwa suatu atribut dapat sekaligus digolongkan pada dua jenis atribut atau lebih. Misalnya, atribut umur merupakan atribut turunan, bernilai tunggal, dan sederhana.

Penentuan suatu atribut apakah sebagai atribut sederhana atau atribut campuran bergantung pada kebutuhan pemakai terhadap data. Misalnya, jika pemakai membutuhkan pencarian alamat pelanggan berdasarkan kode pos atau nama kota, maka atribut alamat pada jenis entitas Pelanggan sebaiknya merupakan atribut campuran. Tetapi jika hal tersebut tidak dibutuhkan maka cukup menetapkan atribut alamat sebagai atribut sederhana.

5.5.1 Langkah Pengidentifikasian

Atribut diidentifikasi dengan membaca dan memahami dokumen spesifikasi kebutuhan pemakai, sebagai berikut.

1. Seperti pada saat pengidentifikasian jenis entitas, atribut diidentifikasi dengan melihat kata benda dan frase kata benda yang merupakan properti, kualitas, pengenal, atau karekteristik dari salah satu jenis entitas atau hubungan entitas pada kamus data.
2. Untuk setiap jenis entitas x dan jenis hubungan entitas y, ajukan pertanyaan "Informasi apa yang diperlukan oleh pemakai terhadap x atau y?". Informasi tersebut merupakan atribut untuk x atau y. Jawaban pertanyaan ini seharusnya dapat ditemukan pada

dokumen spesifikasi, tetapi jika tidak ditemukan, tanyakan langsung kepada pemakai. Misalnya, untuk jenis entitas Mahasiswa dapat ditanyakan kepada pemakai informasi apa saja yang dibutuhkan tentang mahasiswa pada saat aplikasi basis data digunakan.

3. Untuk masing-masing atribut yang telah teridentifikasi tentukan:

   a. Domain, yaitu nilai-nilai yang mungkin untuk atribut tersebut.

   b. Jenis, yaitu salah satu dari enam jenis atribut di atas.

   c. Nilai awalnya jika ada.

   d. Boleh null atau tidak.

4. Dokumentasikan seluruh atribut pada kamus data dengan menyertakan informasi: nama jenis entitas, nama atribut, deskripsi, domain, nama alias, jenis atribut, nilai awal atribut, dan boleh tidaknya null. Tuliskan dalam format tabel agar mudah dipahami. Contoh penulisan kamus data ini dapat dilihat pada Subbab 11.3.

Nama Jenis Entitas	Nama Atribut	Deskripsi	Domain	Alias	Jenis Atribut	Nilai Awal	Null

5.5.2 Notasi UML

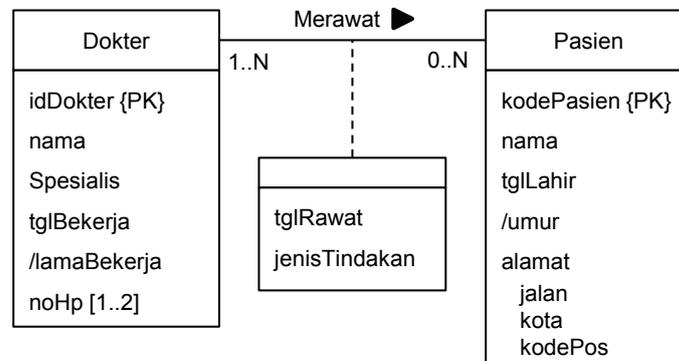
Untuk menuliskan atribut pada jenis entitas, notasi segi empat jenis entitas dibagi menjadi dua bagian. Di mana pada bagian atas adalah nama jenis entitas, dan bagian bawah adalah daftar nama atribut dari jenis entitas tersebut.

Ketentuan penulisan atribut adalah sebagai berikut:

- Awal kata pada nama atribut dituliskan dalam huruf kecil, dan awal kata selanjutnya dalam huruf besar. Misalnya, namaAwal dan namaAkhir.
- Atribut primary key dituliskan pada urutan paling atas diikuti notasi "{PK}". Misalnya, nim {PK}.
- Atribut selain dari primary key dituliskan di bawah atribut kunci.
- Atribut campuran dituliskan berikut dengan nama-nama atribut sederhana penyusunnya.
- Atribut turunan dituliskan dengan menambahkan karakter '/' pada awal nama.
- Atribut bernilai jamak dituliskan dengan menuliskan "[1..n]" pada akhir nama, di mana n adalah jumlah maksimum nilai.
- Atribut sederhana dan atribut bernilai tunggal dituliskan tanpa menambahkan notasi lain.

Atribut untuk jenis hubungan entitas dituliskan sama dengan cara penulisan atribut jenis entitas, hanya saja pada kotak nama entitas dibiarkan kosong, dan di antara atribut tersebut dan jenis hubungan entitas ditarik garis putus-putus.

Contoh 5.10: Penulisan Atribut pada Jenis Entitas & Hubungan Entitas

Gambar 5.9 memperlihatkan contoh penulisan atribut pada jenis entitas Dokter dan Pasien, serta pada jenis hubungan entitas Merawat. Perhatikan bahwa idDokter dan kodePasien masing-masing merupakan primary key, noHP adalah atribut bernilai jamak, alamat adalah atribut campuran, dan lamaBekerja dan umur adalah atribut turunan. ■

Gambar 5.9 Atribut jenis entitas dan hubungan entitas

5.6 GAMBARKAN ERD

Setelah pengidentifikasian jenis entitas, jenis hubungan entitas, dan atribut, langkah selanjutnya adalah menggambarkan ERD lengkap berdasarkan kamus data yang telah dibuat. Jika ERD cukup kompleks, maka cukup dituliskan atribut kunci (primary key) saja pada setiap jenis entitas. Penggambaran ERD dapat menggunakan aplikasi editor UML, misalnya ArgoUML, StarUML, atau Microsoft PowerPoint/Visio[3].

5.7 PERBAIKI STRUKTUR ERD

Setelah pembuatan ERD, langkah selanjutnya adalah memeriksa struktur ERD dan memperbaikinya jika ditemukan permasalahan. Pemeriksaan dilakukan dengan mengidentifikasi keberadaan masalah redudansi dan perangkap pada struktur ERD.

Terdapat dua masalah perangkap: Fan Trap dan Chasm Trap. Kedua masalah ini dapat menimbulkan kesalahpahaman tentang makna objek jenis hubungan entitas sehingga harus dihilangkan.

[3] http://en.wikipedia.org/wiki/List_of_Unified_Modeling_Language_tools menyediakan daftar nama editor UML.

Perbaikan terhadap dua permasalahan struktur ERD dilakukan dengan memodifikasi struktur diagram, baik dengan cara menghilangkan jenis hubungan entitas, menambah jenis hubungan entitas baru, atau mengubah susunan jenis entitas.

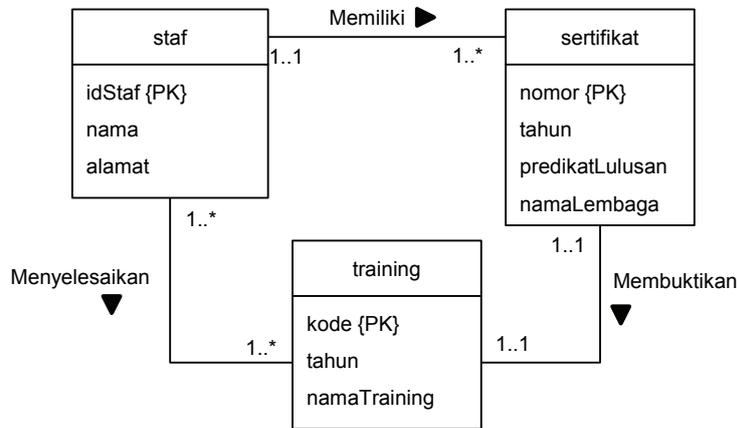
5.7.1 Menghilangkan Redundansi Hubungan

Suatu jenis hubungan entitas dikatakan redundan (berlebihan) jika informasi yang diberikan oleh jenis hubungan entitas tersebut dapat diperoleh melalui jenis hubungan entitas yang lain.

Jenis hubungan entitas yang rendundan, selain menyebabkan ERD tidak minimal (sehingga sulit untuk dipahami), juga menyebabkan redudansi data pada basis data. Redundansi data adalah data yang sama disimpan berulang pada basis data sehingga mengakibatkan pemborosan tempat penyimpanan data. Oleh karena itu, jenis hubungan entitas rendundan harus diidentifikasi dan dihapus dari ERD.

Contoh 5.11: Jenis Hubungan Entitas Redundan

Perhatikan Gambar 5.10. Terdapat dua informasi yang masing-masing dapat diperoleh melalui dua jalur yang berbeda.

1. Informasi training yang telah diselesaikan oleh seorang staf dapat diperoleh secara langsung melalui jenis hubungan entitas Menyelesaikan dan secara tidak langsung melalui jenis hubungan entitas Memiliki dan Membuktikan.
2. Informasi sertifikat yang dimiliki oleh seorang staf dapat diperoleh secara langsung melalui jenis hubungan entitas Memiliki, dan secara tidak langsung melalui jenis hubungan entitas Menyelesaikan dan Membuktikan.

Gambar 5.10 Contoh hubungan entitas yang redundan

Penentuan apakah suatu jenis hubungan entitas redundan dilakukan dengan melihat jenis hubungan entitas mana yang lebih dipentingkan dan/atau yang lebih sering diakses oleh pemakai. Pada contoh ke-2, jika pemakai lebih mementingkan informasi sertifikat seorang staf dan informasi ini lebih sering dibutuhkan, maka jenis hubungan entitas Menyelesaikan adalah redundan dan dapat dihapus. Sebaliknya, pada contoh ke-1, jika pemakai lebih mementingkan info training yang telah diikuti oleh seorang staf dan informasi ini lebih sering diakses oleh pemakai, maka jenis hubungan entitas Memiliki adalah redundan dan dapat dihapus dari diagram.

Perlu diperhatikan bahwa jenis hubungan entitas Membuktikan tidak dapat dihapus dari diagram karena asosiasi di antara sertifikat dan training tidak bisa atau sulit didapat secara tidak langsung dari jenis hubungan entitas Menyelesaikan dan Memiliki. Misalnya, jika seorang karyawan telah mengikuti sejumlah training dan karyawan tersebut memiliki sejumlah sertifikat, maka sulit mengasosiasikan training dan sertifikat jika tidak ada asosiasi yang eksplisit, di antara sertifikat dan training. ■

5.7.2 Menghilangkan Fan Trap

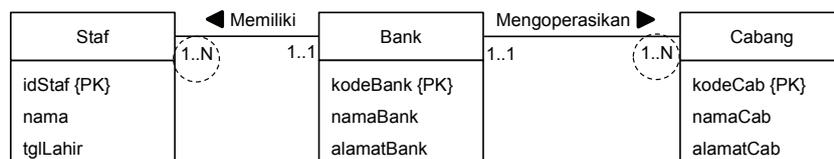
Fan Trap adalah permasalahan struktur ERD yang mengakibatkan terjadinya ketidakpastian hubungan di antara dua objek entitas dari dua jenis entitas yang berbeda.

Fan trap harus dihilangkan dari diagram karena akan mengakibatkan pemberian informasi yang tidak pasti (ambigu) kepada pemakai, pada saat pengambilan data dari basis data. Fan trap terjadi jika memenuhi dua syarat berikut:

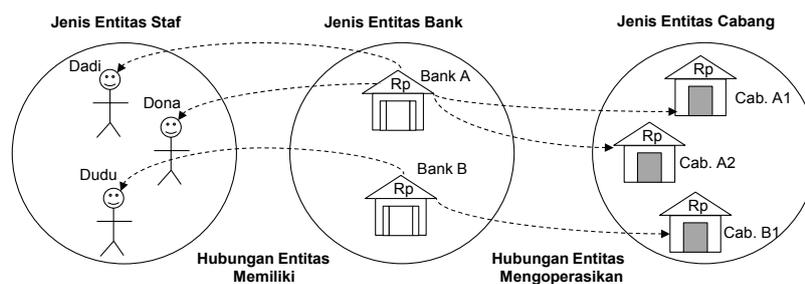
1. Sebuah jenis entitas pusat memiliki hubungan secara langsung ke dua atau lebih jenis entitas lain.
2. Hubungan dari jenis entitas pusat ke dua jenis entitas lain tersebut adalah 1:N.

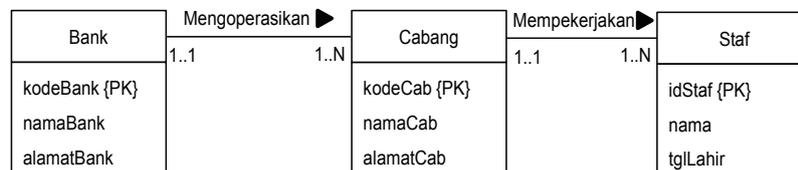
Fan trap dihilangkan dengan merestrukturisasi jenis hubungan entitas sehingga salah satu kondisi di atas tidak terpenuhi.

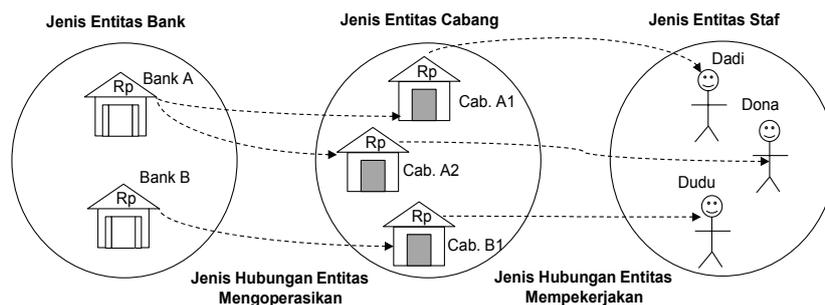
Contoh 5.12: Fan Trap dan Cara Menghilangkannya

Perhatikan ERD bank pada Gambar 5.11. Jenis entitas (pusat) Bank dapat memiliki satu orang staf atau lebih, dan dapat mengoperasikan satu cabang atau lebih. Setiap staf terdaftar hanya pada satu bank saja, dan ditempatkan pada salah satu cabang bank tersebut. Setiap cabang hanya dioperasikan oleh satu bank tertentu saja.

Gambar 5.11 Contoh fan trap pada ERD Bank

Dari gambar dapat dilihat bahwa ERD tersebut memenuhi syarat terjadinya fan trap, yaitu terdapat hubungan 1:N dari jenis entitas Bank ke Staf, dan 1:N dari jenis entitas Bank ke Cabang. Dengan adanya fan trap maka pertanyaan berikut tidak dapat dijawab dengan pasti: "Di cabang manakah seorang staf ditempatkan?".

Untuk melihat ketidakpastian informasi, perhatikan gambar detail objek entitas dan objek hubungan entitas pada Gambar 5.12. Pertanyaan: Di cabang manakah "Dadi" dan "Dona" ditempatkan oleh "Bank A"? Apakah di Cabang "Cab A1" atau "Cab A2"? tidak dapat dijawab dengan pasti, karena "Bank A" memiliki dua cabang, sedangkan seorang staf hanya dapat ditempatkan pada satu cabang.

Gambar 5.12 Objek entitas dan hubungan entitas ERD bank

Permasalahan fan trap pada contoh di atas dapat dihilangkan dengan menghapus jenis hubungan entitas Memiliki di antara jenis entitas Bank dan Staf, dan membuat jenis hubungan entitas baru Mempekerjakan yang menghubungkan jenis entitas Cabang dan Staf (Gambar 5.13).

Gambar 5.13 ERD bank setelah direstrukturisasi

Gambar 5.14 memperlihatkan bahwa dengan melakukan restrukturisasi diagram maka informasi nama cabang untuk masing-masing staf dapat diketahui dengan pasti: "Dadi" ditempatkan di Cabang "Cab A1" dan "Dona" ditempatkan di Cabang "Cab A2". ■

Gambar 5.14 Objek entitas dan jenis hubungan entitas ERD bank setelah direstrukturisasi

5.7.3 Menghilangkan Chasm Trap

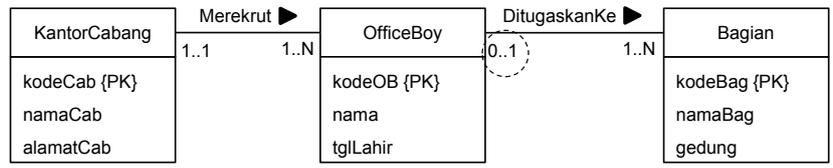
Chasm Trap adalah ketidaknormalan struktur ERD yang mengakibatkan hilangnya hubungan di antara dua objek entitas (dari jenis entitas yang berbeda) yang seharusnya tergambar pada diagram.

Chasm trap harus dihilangkan dari diagram karena mengakibatkan ketidakmampuan basis data untuk memberikan informasi yang seharusnya ada. Chasm trap terjadi jika memenuhi dua syarat berikut:

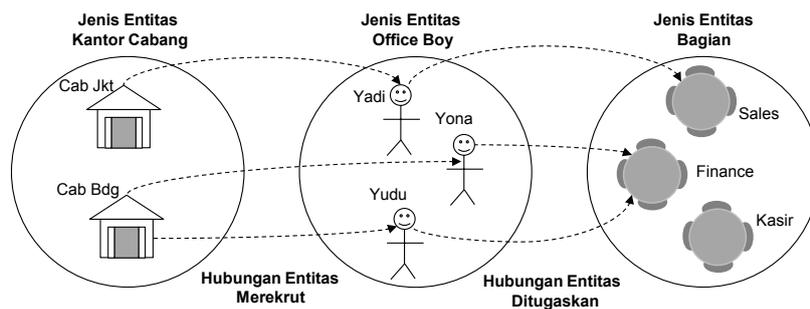
1. Sebuah jenis entitas pusat memiliki hubungan secara langsung ke dua atau lebih jenis entitas lain.
2. Salah satu dari jenis hubungan entitas tersebut dalam arah ke jenis entitas pusat memiliki batasan partisipasi opsional.

Chasm trap dihilangkan dengan menambahkan satu jenis hubungan entitas baru di antara dua jenis entitas yang hubungannya terputus.

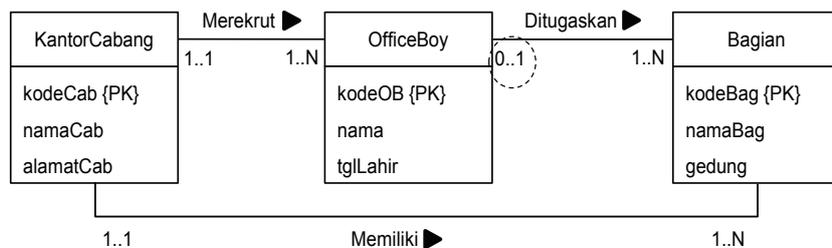
Contoh 5.13: Chasm Trap dan Cara Menghilangkannya

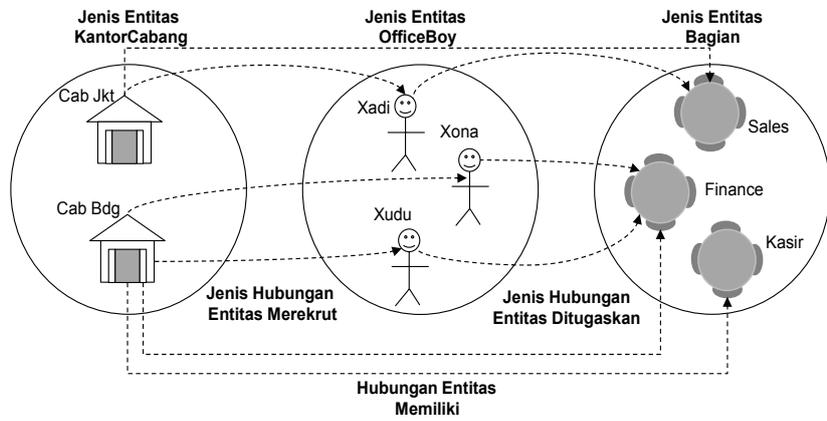
Gambar 5.15 memperlihatkan contoh chasm trap yang terjadi pada sebuah perusahaan yang memiliki beberapa kantor cabang. Setiap kantor cabang merekrut satu office boy (OB) atau lebih, dan setiap OB ditugaskan pada satu bagian atau lebih pada kantor cabang tertentu.

Gambar 5.15 Chasm trap pada ERD kantor cabang

Perlu dicatat bahwa tidak semua bagian memiliki OB, dan oleh karena itu, partisipasi bagian adalah opsional terhadap kepemilikan OB. Akibat dari partisipasi opsional ini, jika suatu bagian tidak memiliki OB maka tidak dapat diketahui di kantor cabang yang mana bagian tersebut berada.

Gambar 5.16 memperlihatkan contoh objek entitas dan hubungan entitas dari ERD pada Gambar 5.15. Kantor Cabang "Jkt" merekrut satu orang OB dan Kantor Cabang "Bdg" merekrut dua orang OB. "Yadi" ditempatkan pada Bagian "Sales" dan "Yona" serta "Yudu" pada Bagian "Finance".

Gambar 5.16 Objek entitas dan hubungan entitas pada ERD cabang perusahaan

Dari diagram tersebut dapat diketahui bahwa Bagian Sales dan Finance masing-masing berada pada Kantor Cabang "Jkt dan Bdg" dengan menelusuri dalam arah berlawanan dari hubungan entitas "Ditugaskan" dan "Merekrut". Berbeda dengan kondisi ini, Bagian Kasir tidak dapat ditetapkan berada pada kantor cabang yang mana, karena tidak ada satu orang OB pun yang ditempatkan pada bagian ini (sehingga jalur terputus).

Permasalahan chasm trap pada contoh di atas dapat dihilangkan dengan menambahkan jenis hubungan entitas (baru) Memiliki di antara jenis entitas KantorCabang dan Bagian (Gambar 5.17).

Gambar 5.17 ERD kantor cabang setelah direstrukturisasi

Gambar 5.18 memperlihatkan bahwa dengan melakukan restrukturisasi diagram Gambar 5.17 maka informasi keberadaan semua cabang (baik yang memiliki OB maupun tidak) dapat ditelusuri secara langsung melalui jenis hubungan entitas Memiliki. Perhatikan bahwa jenis hubungan entitas Memiliki tidak redundan, karena hubungan ini diperlukan sebagai penghubung informasi yang terputus. ■

Gambar 5.18 Objek entitas dan objek hubungan entitas pada ERD kantor cabang setelah direstrukturisasi

5.8 RINGKASAN MINDMAP